数理解析研究所講究録
66巻 1969年 33-42

§3. Semantical Notions

3.1 Quantities

Quantities are abstract elements, and are introduced for describing the course of the elaborations of expressions. Each quantity has its mode, type and value. Let Q be a quantity, then, we shall denote its mode by m(Q), type by t(Q) and value by w(Q). Then it holds

t(Q) = t(m(Q)) = t(w(Q)).

<u>pragmatics</u>

Let V be a <variable>, and E be an <expression>. In a course of a normal program, if V has its ability "able", then V has its quantity denoted by q(V). As the result of the elaboration of E, we shall obtain a quantity Q' or a <label> L. For describing such conclusion, we use the notation

e(E) => Q'

or

e(E) => L

respectively. <u>end of pragmatics</u>

3.2 Values

Values are classified according to their types (or their styles) as follows:

3.2.1 <u>effect</u> type.

There is a sole value <u>done</u> in the <u>effect</u> type.

3.2.2 <u>real</u> type.

A value of the <u>real</u> type is a real number. We shall use the following notations:

$\mathbb{R}$: the set of all real numbers.

$\mathbb{I}$: the set of all integers, in the sense of the subset of $\mathbb{R}$.

round(R) : the integer obtained by rounding R, where R is a real number. (round(R) = entier(R+0.5).)

3.2.3 <u>bits</u> type.

A value of the <u>bits</u> type is a bit-string. Bit-strings are defined with its length ($\in \mathbb{I}$), recursively as follows:

1) ε is a bit-string of length 0.
2) <u>0</u> is a bit-string of length 1.
3) <u>1</u> is a bit-string of length 1.
4) Let B be a bit-string of length n (≥ 1). B<u>0</u> is a bit-string of length n+1.
5) Let B be a bit-string of length n (≥ 1). B<u>1</u> is a bit-string of length n+1.

We shall use the following notations:

$\mathbb{B}$: the set of all bit-strings.

$\mathbb{B}_I$: the set of all bit-strings of length I, where I is an integer (≥ 0).

length(B) : length of B, where B is a bit-string.

3.2.4 <u>string</u> type.

A value of the <u>string</u> type is a <string>. <string>'s are defined with its length ($\in \mathbb{I}$), recursively as follows:

1) ‘’ is a <string> of length 0.
2) Let n be an integer (≥ 1); and A_i be a <basic symbol> other than ‘ and ’, or a <string>, and if A_i is a <basic symbol> then let m_i stand for 1, if A_i is a <string> of length m then let m_i stand for m+2, for (i=1,2,....,n); then

   ‘$A_1 \ldots . A_n$’

   is a <string>of length $m_1+m_2+\ldots.+m_n$.

We shall use the following notations:

$\mathbb{C}$: the set of all <string>'s.

$\mathbb{C}_I$: the set of all <string>'s of length I, where I is an integer (≥ 0).

length(C) : the length of C, where C is a <string>.

3.2.5 reference type.

A value of the reference type is the empty set Ø or a set {Q} with a sole element Q, where Q is a quantity.

3.2.6 array style.

Let T be a type of the form

array T'

where T' is a type.

1) The empty set Ø is a value of type T.

2) Let v be an integer,

u be an integer ($\geq v$), and

Q_i be a quantity of type T', for i=v,v+1,....,u.

Then the set

$\{<v,Q_v>,<v+1,Q_{v+1}>,....,<u,Q_u>\}$

is a value of type T. (<v,Q> denotes the ordered pair of v and Q.)

3.2.7 structure style.

3.2.7.1 Let T be a type of the form

structure $(S_1T_1,....,S_nT_n)$

where n is an integer (≥ 1);

S_i is a <selector> for i=1,2,....,n;

T_i is a type for i=1,2,....,n.

Let Q_i be a quantity of type T_i, for i=1,2,....,n. Then the set

$\{<S_1,Q_1>,<S_2,Q_2>,....,<S_n,Q_n>\}$

is a value of type T.

3.2.7.2 Let T be a type of the form

structure ().

There is a sole value, the empty set Ø, in the type T.

3.2.8 procedure style.

Let T be a type of the form

procedure $(T_1,\ldots,T_n)T'$

where n is an integer (≥ 0);

T_i is a type for $i=1,2,\ldots,n$;

T' is a type.

Let V_i be a <variable> different from each other, for $i=1,2,\ldots,n$,

and let E be an <expression>; then

$(V_1,\ldots,V_n)E$

is a value of type T.

3.3 Modes

Modes and their types are defined recursively as follows:

1) effect is a mode of type effect.

2.1) Let R_i be a real number for $i=1,2,3$, then

real$[R_1:R_2:R_3]$

is a mode of type real.

2.2) Let R be a real number, then

real [precision R]

is a mode of type real.

3.1) Let I be an integer, then

bits [exact I]

is a mode of type bits.

3.2) Let I be an integer, then

bits [varying I]

is a mode of type bits.

4.1) Let I be an integer, then

string [exact I]

is a mode of type string.

4.2) Let I be an integer, then

<u>string</u> <u>[varying I]</u>

is a mode of type string.

5) <u>reference</u> is a mode of type <u>reference</u>.

6) Let I_i be an integer for i=1,2; and let T be a type; then

<u>array</u> $[I_1:I_2]T$

is a mode of type <u>array</u> T.

7) Let T be a type of structure style, then T is a mode of type T.

8) Let T be a type of procedure style, then T is a mode of type T.

A mode specifies a domain of values. Let M be a mode. The domain of values specified by M is denoted by

$\mathbb{W}(M)$,

and is defined as follows:

3.3.1 $\mathbb{W}$(<u>effect</u>) is {<u>done</u>}.

3.3.2.1 Let R_i be a real number for i=1,2,3. Then, $\mathbb{W}$ (<u>real</u> $[R_1:R_2:R_3]$) is the finite set

$\{x \mid x \in \mathbb{R} \wedge R_1 \leq x \wedge x \leq R_3 \wedge$ there exist an integer y such that $x = y \times R_2\}$.

3.3.2.2 Let R be a real number. Then $\mathbb{W}$ (<u>real</u> <u>[precision</u> R]) is some finite set $\mathbb{W}$ of real numbers which satisfies following conditions:

a) If $0 \neq x \in \mathbb{W}$ and $0 \neq y \in \mathbb{W}$ and $x<y$ and there are no element z of such that $x<z<y$, then

$y-x<\frac{1}{2}(|x|+|y|)\times|R|$.

b) There exists a positive number in $\mathbb{W}$ with a sufficiently large absolute value.

c) There exists a negative number in $\mathbb{W}$ with a sufficiently large absolute value.

d) There exists a positive number in $\mathbb{W}$ with a sufficiently small absolute value.

e) There exists a negative number in $\mathbb{W}$ with a sufficiently small absolute value.

(The meaning of the adverb "sufficiently" is unspecified.)

3.3.3 Let I be an integer.

$\mathbb{W}$(<u>bits</u> <u>[exact</u> I<u>]</u>)is $\mathbb{B}_I$ if $I \geq 0$,

$\emptyset$ if $I<0$.

$\mathbb{W}$(<u>bits</u> <u>[varying</u> I<u>]</u>)is $\mathbb{B}_0 \cup \mathbb{B}_1 \cup \ldots \cup \mathbb{B}_I$ if $I \geq 0$,

$\emptyset$ if $I<0$.

3.3.4 Let I be an integer.

$\mathbb{W}$(<u>string</u> <u>[exact</u> I<u>]</u>) is $\mathbb{C}_I$ if $I \geq 0$,

$\emptyset$ if $I<0$.

$\mathbb{W}$(<u>string</u> <u>[varying</u> I<u>]</u>) is $\mathbb{C}_0 \cup \mathbb{C}_1 \cup \ldots \cup \mathbb{C}_I$ if $I \geq 0$,

$\emptyset$ if $I<0$.

3.3.5 $\mathbb{W}$(<u>reference</u>) is $\{\emptyset\} \cup \{\{Q\} \mid Q \in \mathbb{Q}\}$.

3.3.6 Let I be an integer, I' be an integer, and let T be a type.

Then $\mathbb{W}$(<u>array</u> <u>[</u>I:I'<u>]</u>T) is

$\{\{<I,Q_I>,<I+1,Q_{I+1}>,\ldots,<I',Q_{I'}>\} \mid Q_i \in \mathbb{Q} \wedge t(Q_i)=T,$

for $i=I,I+1,\ldots,I'\}$ if $I \leq I'$,

$\emptyset$ if $I>I'$.

3.3.7 Let n be an integer (≥ 0); S_i be a <selector> different from each other, and T_i be a type for $i=1,2,\ldots,n$. Then

$\mathbb{W}$(<u>structure</u> <u>(</u>$S_1T_1,\ldots,S_nT_n$<u>)</u>) is

$\{\{<S_1,Q_1>,<S_2,Q_2>,\ldots,<S_n,Q_n>\} \mid Q_i \in \mathbb{Q} \wedge t(Q_i)=T_i$,...

for i=1,2,....,n}

3.3.8 Let n be an integer (≥ 0); T_i be a type for i=1,2,....,n; and T be a type. Then $\mathcal{W}$(procedure (T_1,....,T_n)T) is the set

$\{(V_1,\ldots,V_n)E \mid V_i$ is <variable>for i=1,2,....,n $\wedge$

E is <expression> without <mark> $\wedge$

"begin let V_1 be T_1;

....

let V_n be T_n;

E end"

is a legal <expression>}.

3.4 Implementation Dependent Factors

When we are concerned with a particular implementation, it is usual that not all values are realized in the implementation. So, the domain of values may be restricted, and biassed in the form of implementation dependent. In the following , we use the notation $\mathcal{W}_M$ for such an implementation dependent set, transformed from $\mathcal{W}$(M). In each implementation, modes are classified by the coincidence of the set $\mathcal{W}_M$. And we shall denote the representative of the class, which contains a mode M, by d(M).

We shall use the following notations:

R1: An (implementation dependent) fixed negative real number with sufficiently large absolute value. It acts as a proxy in a <real modifier> of the form [E_1:E_2:E_3] when E_1 is absent.

R2: An (implementation dependent) fixed positive real number with sufficiently large absolute value. It acts as a proxy in a <real modifier> of the form [E_1:E_2:E_3] when E_3 is absent.

R3: An (implementation dependent) fixed positive real number with sufficiently small absolute value. It acts as a proxy in a <real

modifier> of the form [precision F] when F is absent.

I1: An (implementation dependent) fixed positive integer. It acts as a proxy in a <bits modifier> of the form [exact I] when I is absent.

I2: An (implementation dependent) fixed positive integer (usually $\geq$I1). It acts as a proxy in a <bit modifier> of the form [varying I] when I is absent.

I3: An (implementation dependent) fixed positive integer. It acts as a proxy in a <string modifier> of the form [exact I] when I is absent.

I4: An (implementation dependent) fixed positive integer (usually $\geq$I3). It acts as a proxy in a <string modifier> of the form [varying I] when I is absent.

3.5 Projections

When a value is assigned for a quantity, it is adjusted (or rounded) for the quantity's mode. Let W be a value, and let M be a mode of the same type with W. We shall denote such an adjusted value by p(M,W) or $p_M(W)$, and call it "the projection of W for M". Obviously it suffices that $p_M(W) \in \mathcal{W}_M$, and $p_M(W)$ is not defined if $\mathcal{W}_M = \emptyset$. Since the set $\mathcal{W}_M$ is implementation dependent, p_M is implementation dependent, too. So the following directions are not compulsory, though the implementors and users are suggested to refer to it.

3.5.1 p(effect,done) is done.

3.5.2 Let M be a mode of real type, and let R be a real number. Then p(M,R) is a real number in $\mathcal{W}_M$ which is nearest to R.

3.5.3 Let I be an integer ($\geq$0), and let B be a bit-string. Then p(bits [exact I],B) is

(ε if I=0;

$$\begin{cases} \underbrace{\underline{0}\ldots\underline{0}}_{I} \text{ if } I>0 \text{ and } B \text{ is } \varepsilon; \\ b_1\ldots b_I \text{ if } I>0 \text{ and } B \text{ is } b_1\ldots b_n \text{ where } n=\text{length}(B)\geq I, \text{ and} \\ \quad b_i \text{ is } \underline{0} \text{ or } \underline{1} \text{ for } i=1,2,\ldots,n; \\ b_1\ldots b_n\underbrace{\underline{0}\ldots\underline{0}}_{I-n} \text{ if } I>0 \text{ and } B \text{ is } b_1\ldots b_n \text{ where } n=\text{length}(B)<I, \\ \quad \text{and } b_i \text{ is } \underline{0} \text{ or } \underline{1} \text{ for } i=1,2,\ldots,n. \end{cases}$$

p(<u>bits</u> <u>[varying</u> I<u>]</u>,B) is

$$\begin{cases} \varepsilon \text{ if } I=0 \text{ or } B \text{ is } \varepsilon; \\ b_1\ldots b_I \text{ if } I>0 \text{ and } B \text{ is } b_1\ldots b_n \text{ where } n=\text{length}(B)\geq I, \text{ and} \\ \quad b_i \text{ is } \underline{0} \text{ or } \underline{1} \text{ for } i=1,2,\ldots,n; \\ B \text{ if } I>0 \text{ and length}(B)<I. \end{cases}$$

3.5.4 Let I be an integer (≥ 0), and let C be a <string> of the form

‘$c_1c_2\ldots c_n$’

where n=length(C), c_i is a <basic symbol> for i=1,2,....,n.

Then p(<u>string</u> <u>[exact</u> I<u>]</u>,C) is

$$\begin{cases} \text{‘}c_1\ldots c_I\text{’ if } n\geq I, \\ \text{‘}c_1\ldots c_n\underbrace{\sqcup\ldots\sqcup}_{I-n}\text{’ if } n<I. \end{cases}$$

p(<u>string</u> <u>[varying</u> I<u>]</u>,C) is

$$\begin{cases} \text{‘}c_1\ldots c_I\text{’ if } n\geq I, \\ C \text{ if } n<I. \end{cases}$$

3.5.5 Let W be a value of <u>reference</u> type.

Then p(<u>reference</u>,W) is W.

3.5.6 Let I be an integer, I' be an integer $\geq$I, and let T be a type.

And let W be a set of the form

$\{<v,Q_v>,<v+1,Q_{v+1}>,\ldots,<u,Q_u>\}$,

where Q_i is a quantity of type T.

Then p(<u>array</u> <u>[</u>I:I'<u>]</u>T,W) is

$\{<I,Q_I'>,<I+1,Q_{I+1}'>,\ldots.,<I',Q_{I'}'>\}$

where if $v \leq j \leq u$ then Q_j' is Q_j, else Q_j' is unspecified, (but $t(Q_j')$ is T) for $j=I,I+1,\ldots.,I'$.

3.5.7 If t(M) is structure style, and t(W) coincides with t(M), then p(M,W) is W.

3.5.8 If t(M) is procedure style, and t(W) coincides with t(M), then p(M,W) is W.

3.6 Abilities

During the elaboration, each <variable> is in a state of ability which is either <u>able</u> or <u>inable</u>, and may be changed to its alternative. We shall denote such ability of a <variable> V by a(V). If a <variable> V is <u>able</u>, V is associated with some quantity, which we shall denote by q(V).

<u>pragmatics</u>

Briefly speaking, a <variable> is made <u>inable</u> at the entrance of the <block>, and is made <u>able</u> at the end of the elaboration of its <declaration>. <u>end of pragmatics</u>